

## IN THE UNITED STATES PATENT and TRADEMARK OFFICE

Inventors:	Yigal Mordechai Edery,	)	
	Nimrod Itzhak Vered, David R.	)	
	Kroll, Shlomo Touboul	)	Control No.: Unassigned
		)	
Patent No.:	7,647,633	)	
		)	
Issue Date:	Jan. 12, 2010	)	
		)	
Filing Date:	June 22, 2005	)	
		)	
Title:	MALICIOUS MOBILE CODE	)	
	RUNTIME MONITORING	)	
	SYSTEM AND METHODS	)	

Mail Stop Ex Parte Reexam  
 Central Reexamination Unit  
 Office of Patent Legal Administration  
 United States Patent & Trademark Office  
 P.O. Box 1450  
 Alexandria, VA 22313-1450

**ATTACHMENT TO REQUEST FOR EX-PARTE REEXAMINATION (FORM PTO-SB/57; PTO-1465) PROVIDING INFORMATION ON U.S. PATENT NO. 7,647,633**

Reexamination under 35 U.S.C. §§ 302-307 and 37 C.F.R. § 1.510 is respectfully requested of United States Patent No. 7,647,633 (the “Edery 633 patent”), which was filed on June 22, 2005 and issued on January 12, 2010. The Edery 633 patent is enforceable and reexamination is appropriate under 37 C.F.R. § 1.510(a). The Edery 633 patent currently is being asserted in several patent infringement cases: *Finjan, Inc. v. FireEye, Inc.*, 13-cv-3133 (N.D. Cal.), *Finjan v. Blue Coat Systems, Inc.*, 13-cv-3999 (N.D. Cal.), and *Finjan v. Websense, Inc.*, 13-cv-4398 (N.D. Cal.).

**I. CLAIMS FOR WHICH REEXAMINATION IS REQUESTED**

Reexamination is requested of claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32 and 33 of the Edery 633 patent.

**II. CITATION OF PRIOR ART**

The Edery 633 patent was filed on June 22, 2005 as application No. 11/159,455 (the “455 application”). It claims priority to U.S. Patent No. 7,058,822 to Edery (“Edery 822”), which was filed on May 17, 2001.

Requester seeks reexamination of the Edery 633 patent on three separate grounds, each of which raises substantial new questions of patentability (“SNQP”). The bases for the request for reexamination primarily lie in the lack of substantive consideration of the three references, either alone or in combination. Because this request presents these three references in a manner not substantively considered by the Office before, reexamination of the Edery 633 patent is warranted.

Reexamination is first requested in light of U.S. Patent No. 5,983,348 to Ji (the “Ji patent”). The Ji patent was filed on September 10, 1997 and issued on November 9, 1999. Accordingly, the Ji patent is prior art to the Edery 633 patent under 35 U.S.C. § 102(e). The Ji patent discloses all elements of the Edery 633 patent’s claims, specifically claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32 and 33. As more thoroughly discussed below, the Ji patent was not substantively considered during the examination of the Edery 633 patent, although it was cited by the examiner.

Reexamination also is requested in light of the combination of the Ji patent and U.S. Patent No. 6,058,482 to Liu (the “Liu patent”). The Liu patent was filed on May 22, 1998. Accordingly, the Ji patent in combination with the Liu patent serves as prior art under 35 U.S.C. § 103(a) to the Edery 633 patent. Together, the combination of the Ji patent and the Liu patent discloses all elements of the Edery 633 patent’s claims, specifically claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32 and 33. Importantly, the Liu patent was neither cited nor referenced during the prosecution of the Edery 633 patent. Accordingly, the combination of the Ji patent with the Liu patent provides an entirely new ground of invalidity not considered by the examiner.

Reexamination also is requested in light of the combination of the Ji patent and U.S. Patent No. 5,974,549 to Golan (the “Golan patent”). The Golan patent was filed on March 27, 1997. Accordingly, the Golan patent in combination with the Ji patent qualifies as prior art under 35 U.S.C. § 103(a) to the Edery 633 patent. The combination of the Golan patent with the Ji patent was not substantively considered during the examination of the Edery 633 patent, even though the Golan patent itself was disclosed in the specification by the patentee and considered by the examiner. Because the combination of the Ji patent and the Golan patent was not considered by the examiner, these patents provide a third new ground for examination.

### III. STATEMENT POINTING OUT SUBSTANTIAL NEW QUESTIONS OF PATENTABILITY

The three bases for reexamination discussed above (the Ji patent alone, the Ji patent in combination with the Liu patent, and the Ji patent in combination with the Golan patent) each establish a substantial new question of patentability of claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32 and 33 of the Edery 633 patent. These substantial new questions of patentability meet the legal standard for ordering *ex parte* reexamination as set forth in the MPEP § 2216:

It must first be demonstrated that a patent or printed publication that is relied upon in a proposed rejection presents a new, non-cumulative technological teaching that was not previously considered and discussed on the record during the prosecution of the application that resulted in the patent for which reexamination is requested, and during the prosecution of any other prior proceeding involving the patent for which reexamination is requested.

The Ji patent, the Ji patent in combination with the Liu patent, and the Ji patent in combination with the Golan patent discloses the requisite elements in the claims of the Edery 633 patent for which reexamination is requested. Therefore, the Office should grant this request.

#### A. Background of the Edery 633 Patent

Generally, the Edery 633 patent relates to a computer processor-based method that, according to claim 1, includes: (1) a computer receiving downloadable information, (2) the computer determining whether the downloadable information includes executable code, and (3) if the downloadable information includes executable code, transmitting mobile protection code from the computer to another device.

More specifically, the Edery 633 patent relates to protection systems and methods capable of protecting network accessible devices or processes from malicious operations. The disclosed embodiments provide for determining whether received information from a third party includes executable code. The Edery 633 patent provides for determining, within one or more network servers, whether a received downloadable includes executable code. Additionally, the embodiments provide for a protection engine that operates within one or more network servers, firewalls, or other network connectable information devices. The protection engine itself includes an information monitor for monitoring the information received by the server. A code detection engine determines whether any of the information received by the server includes executable code.

Furthermore, the embodiments disclosed in the Edery 633 patent contemplate “delivering static, configurable and/or extensible remotely operable protection policies to a Downloadable-destination, more typically as a sandboxed package including the mobile protection code, downloadable policies and one or more received Downloadables.” Edery 633 at Col. 2, lines 45-49. The embodiments further disclose causing the mobile protection code to be executed within the destination, which could include a web browser, in a manner that enables the downloadable operations to be detected, intercepted, or further responded to via the various protected operations. *See e.g.*, Edery 633 at Col. 2 lines 39-57.

The methods disclosed in claims 1 and 28 are depicted in Figure 9 of the Edery 633 patent:

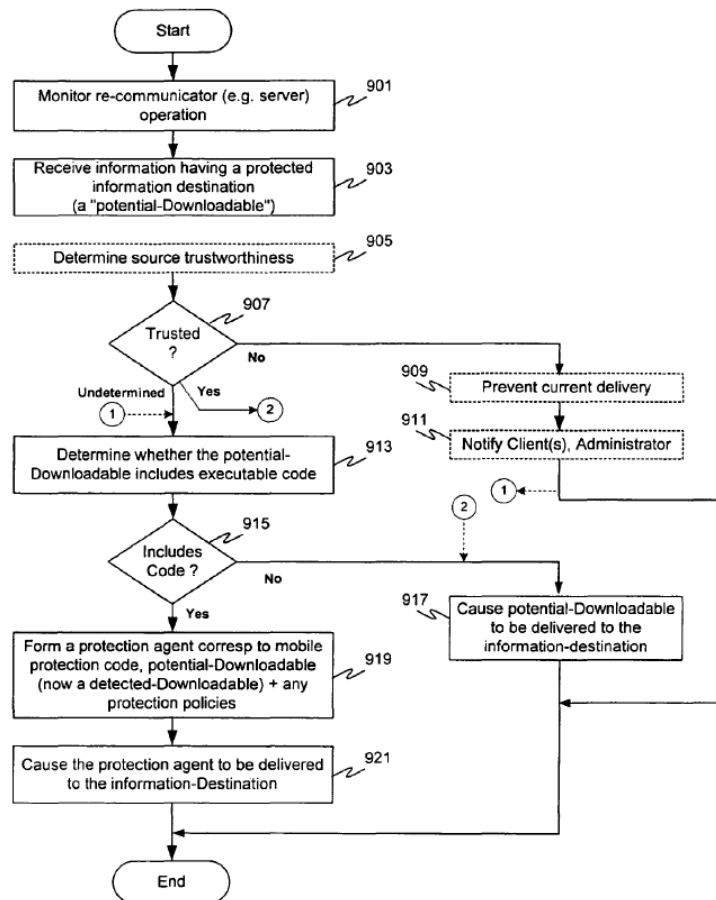


FIG. 9

The methods involve a computer receiving downloadable information. Also, the methods provide for the sandboxed package that includes mobile protection code to be received by the computer at a downloadable destination. The computer determines whether the downloadable

information includes executable code. If executable code exists, mobile protection code is transmitted from the computer to a third information destination device.

#### **B. The Examination of the Edery 633 Patent**

The Edery 633 patent was prosecuted as the 455 application. On February 25, 2009, all pending claims were rejected as unpatentable based on non-statutory obviousness-type double patenting over claims 1-35 of the Edery 822 patent. The rejection stated that although the conflicting claims were not identical, they were not patentably distinct from one another. The examiner found the pending claims anticipated by claims 1-35 of the Edery 822 patent because the claims of the Edery 822 patent contained all the limitations present in the 455 application. The examiner thus concluded that the pending claims of the 455 application were not patentable under the doctrine of obvious-type double patenting.

In the same Office Action, the examiner also rejected all pending claims under 35 U.S.C. § 101 as directed to non-statutory subject matter. The examiner found that the pending claims failed to (1) be tied to a particular machine or (2) transform underlying subject matter (such as an article or material) to a different state or thing. The examiner stated that the pending claims of the 455 application lacked a positive tie to a particular machine that accomplished the claimed method steps or transformed underlying subject matter.

The examiner also rejected all pending claims under 35 U.S.C. § 102(e) as anticipated by the Golan patent. For pending independent claims 1, 16, 28 and 29 of the 455 application, the examiner found that the Golan patent anticipated the claims of the 455 application because the Golan patent discloses:

a method, system, and a computer readable storage medium storing computer code for causing a computer to receive downloadable information by a security (information) monitor, determine whether the downloadable information includes executable code as determined by a security monitor (content inspection engine) that is communicatively coupled to the security (information) monitor, and a sandbox (protection agent engine) communicatively coupled to the security monitor (content inspection engine) for causing mobile protection code to be communicated to one information-destination of downloadable information, if the downloadable information is determined to include executable code.

February 25, 2009 Office Action at 4. The examiner found that these elements anticipated pending claims 1, 16, 28, and 19 of the 455 application as disclosed in Golan col. 2, lines 12-28; col. 3, lines 45-58; and col. 4, line 50 through col. 5, line 14.

Following the February 25 Office Action, on May 26, 2009, the owner and assignee of the 455 application and Edery 822, Finjan, Inc., filed a terminal disclaimer disclaiming any statutory term by a patent that would mature from the 455 application beyond the expiration date of the Edery 822 application.

Also on May 26, 2009, the applicants filed an amendment and response to the February 25 Office Action. The amendment revised the specification to reference that the 455 application is a continuation of Edery 822, as well as incorporating by reference a number of other patents and applications. May 26, 2009 Amendment at 2. In an attempt to overcome the § 101 rejections, the applicants further amended the pending claims to include a computer/computer usable medium as the required machine under § 101. *Id.* at 3-14.

When the applicants addressed the rejections under § 102(e) to Golan, they emphasized the packaging of protection code in the 455 application. Golan was distinguishable, according to the applicants, because it focuses on when the security monitor is already resident on a client computer and does not concern itself with how the security monitor is installed. *Id.* at 15. The applicants described the 455 application's claimed improvements:

[P]rima facie the methodology of the claimed invention, of packaging mobile protection code with downloadable information, seems wasteful and counter-intuitive, since such protection code is typically re-transmitted to the client computer many times – in particular, each time a downloadable with execution code is downloaded. However, the advantage of this methodology is control over the ability to customize the mobile protection code and to update it as necessary, thus obviating the need for a user to be responsible for ensuring that his security code be appropriate to his computer and up to date.

*Id.* at 15. The applicants then incorporated that argument for all pending claims of the 455 application discussed in the remarks, stating that “Golan does not describe causing mobile protection code, which corresponds to Golan's security monitor, to be communicated.” *Id.* at 15-16. The applicants further included arguments as to each dependent claim that the mobile code being communicated did not exist in the Golan patent's disclosures. *Id.* at 16-20. Thus, the

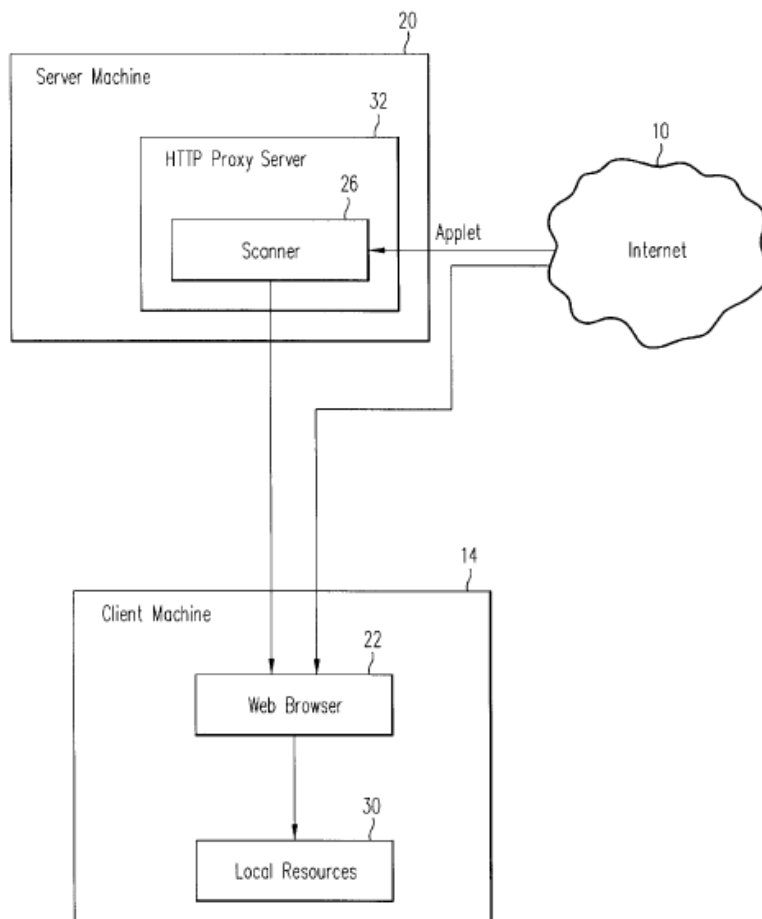
primary argument by the applicants was that the Golan patent failed to disclose the technique commonly referred to as “sandboxing.”

On June 26, 2009, all claims were allowed without comment. On July 2, 2009, applicants filed a Request for Continued Examination and disclosed a list of references for review and citation on the 455 application. On July 15, 2009, in response to the Request for Continued Examination, the examiner considered the list of references submitted by the applicants and allowed all claims without substantive comment, only removing inappropriate hyperlinks. On January 12, 2010, the 455 application issued as the Edery 633 patent.

**C. SNQP 1 - the Ji patent raises a SNQP as to Claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32, and 33 under 37 CFR 1.510(b)(1)**

As discussed above, during prosecution of the Edery 633 patent, the applicants argued that the Golan patent failed to disclose the mobile protection code to be communicated. Following this argument from the applicants, the examiner allowed the 455 application to issue as the Edery 633 patent without comment. From the examiner’s silence, it can be inferred that the alleged failure of Golan to disclose the mobile protection code served as the basis for allowance. As detailed below, however, the Ji patent teaches creating a sandboxed package including mobile protection code, the downloadable-information and the security policies, where that sandboxed package is subsequently communicated to the intended client destination—the elements of the Edery 633 patent including the allegedly missing element from the prior art. Had the applicants informed the examiner that the mobile protection code was communicated to the destination in the Ji patent, the examiner would have not allowed the claims as issued. Accordingly, the Ji patent’s disclosures raise a SNQP because of its disclosure of the elements claimed by the Edery 633 patent, and because the prosecution of the Edery 633 patent did not include a substantive review of the Ji patent. *See In re Swanson*, 540 F.3d 1368, 1380 (Fed. Cir. 2008) (“the PTO should evaluate the context in which the reference was previously considered and the scope of the prior consideration and determine whether the reference is now being considered for a substantially different purpose”).

Like the Edery 633 patent, the Ji patent also relates to a computer processor-based method that includes (1) a computer receiving downloadable information, (2) the computer determining whether the downloadable information includes executable code, and (3) if the downloadable information includes executable code, transmitting mobile protection code from the computer to another device. Figure 1 of the Ji patent is shown below:



The Ji patent discloses feature (1), a computer receiving downloadable information (the Applet) at server machine 20, as shown in Figure 1 above and at column 4, lines 55-65. Feature (1) is additionally found in column 3 lines 19-22: “The applets or controls (hereinafter collectively referred to as applets) are conventionally received from e.g. the Internet or an Intranet at a conventional server.” Accordingly, the Ji patent discloses the first element requiring a computer receiving downloadable-information and therefore raises a SNQP.

The Ji patent also discloses feature (2), “determining, by the computer, whether the downloadable-information includes executable code,” by scanning applets and identifying “suspicious instructions” in the downloadable-information for instrumentation. *See* Ji at 3:16-31; 4:66-5:4. The search of suspicious operations in Ji is analogous to determining whether the downloadable-information includes executable code. Accordingly, the Ji patent discloses the feature claimed in the Edery 633 patent that the computer determines whether the downloadable information includes executable code and therefore raises a SNQP.

The Ji patent also discloses feature (3) which requires that, if the downloadable information includes executable code, transmitting the mobile protection code from the computer to another device. The Ji patent discloses feature (3) through the disclosure of the monitoring package being delivered to the client. *See* Ji at 3:32-56; 6:38-51. This is particularly evidenced by the Ji patent's disclosure of "pre and post filter monitoring package security policy functions) (sic) are combined with the instrumented applet code in a single JAR (Java archive) file format at the server 32, and downloaded to the web browser 22 in client machine 14." Ji at 6:38-42. Additionally, this is found where "[a]s the applet code is executed, each instrumented instruction is monitored by the web browser using a monitor package which is part of the scanner and delivered to the client side." Ji at 3:35-39. Accordingly, the Ji patent discloses the feature claimed by Edery 633 that if the downloadable includes executable code, the mobile protection code is transmitted from the computer to another device and therefore raises a SNQP.

The Ji patent additionally discloses all four features of claim 28 that include (a) receiving a sandboxed package that includes mobile protection code and a downloadable and one or more protection policies at a computer, (b) the mobile protection code on the computer causing one or more operations attempted by the downloadable to be received by the mobile protection code, (c) the mobile protection code receiving an attempted operation of the downloadable, and (d) the mobile protection code initiating a protection policy corresponding to the attempted operation.

The Ji patent discloses feature (a) of claim 28 in the "pre and post filter monitoring package security policy functions) (sic) are combined with the instrumented applet code in a single JAR (Java archive) file format at the server 32, and downloaded to the web browser 22 in client machine 14". Ji at 6:38-42. Accordingly, the Ji patent raises a SNQP as to feature (a) of claim 28.

The Ji patent also discloses feature (b) of claim 28 requiring mobile protection code on the computer causing operations attempted by the downloadable to be received by the mobile protection code. The Ji patent discloses this feature through its discussion of the monitoring done by the web browser and "[a]ll the monitoring and applet code is executed in the web browser 22 in the client machine 14." *See* Ji at 6:38-51. Accordingly, the Ji patent raises a SNQP as to feature (b) of claim 28.

The Ji patent also discloses feature (c) of claim 28 which requires the mobile protection code receiving an attempted operation of the downloadable. The Ji patent discloses this through

its description where the “applet code is executed, each instrumented instruction is monitored by the web browser using a monitor package which is part of the scanner and delivered to the client side.” Ji at 3:35-38. Accordingly, the Ji patent raises a SNQP as to feature (c) of claim 28.

The Ji patent discloses feature (d) of claim 28 requiring the mobile protection code to initiate a protection policy corresponding to the attempted operation. This disclosure occurs in the Ji patent’s discussion that “[i]f the security policy . . . is violated, that particular instruction which violates the security policy is not executed, and instead a report is made and execution continues, if appropriate, with the next instruction.” Ji 3:40-44. Accordingly, the Ji patent raises a SNQP as to feature (d) of claim 28.

Because claims 2, 3, 4, 5, 6, and 7 are dependent upon independent claim 1 and claims 29, 30, 31, 32, and 33 are dependent upon independent claim 28, a SNQP is raised as to each of the dependent claims as well.

In the prosecution file history, the applicants distinguished the prior art by emphasizing the 455 application’s focus on the packaging of protection code, whereas the prior art focused on a situation where the security monitor is already resident on a client computer while failing to concern itself with how the security monitor was installed. However, the Ji patent discloses the very packaging and communication of mobile protection code used to overcome the examiner’s rejections. To the extent the patentee argues that the Ji patent was discussed during prosecution, the Ji patent was not considered by the examiner for it is disclosed ability of communicating mobile protection code from the server to the client computer. Furthermore, future suggestions by patentee that the Ji patent’s disclosure of both static and dynamic Java applet scanning is distinguishable from the Edery 633 patent would be a red herring attempt to distract the examiner with terminology that is not relevant to the identified claims. Accordingly, the Ji patent raises a SNQP as to claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32, and 33.

**D. SNQP 2 – The Ji patent in view of the Liu patent raises a SNQP as to Claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32, and 33 under 37 CFR 1.510(b)(1)**

The articulated *KSR International Co. v. Teleflex Inc.* obviousness standard<sup>1</sup> dictates that the teachings from the Ji and Liu patents related to the system and methods for protecting

---

<sup>1</sup> In *KSR International Co. v. Teleflex Inc.*, 550 U.S. 398, 415 (2007), the Supreme court “beg[an] by rejecting the rigid approach of the Court of Appeals (i.e., requiring satisfaction of the “teaching, suggestion, motivation” (TSM) test) to show an invention would have been obvious (and is therefore unpatentable). Returning to its own nonobviousness cases, the Court

network-connectible devices from undesirable downloadable operations are properly combinable and representative of the obvious body of knowledge well within the grasp of the average practitioner skilled in the art of computer network protection. Therefore, the prior art references, Ji and Liu, are not limited to Java functionality but encompass a broader base of computer technology. One of ordinary skill in the art would be motivated to include the additional disclosures from the Liu patent, because the Liu patent, like the Ji patent, addresses network and computer security, including stopping viruses and other malicious software attacks resulting from the use of network languages like Java. *See* Liu at 1:40-2:58, 3:10-18, 3:31-40, 4:28-50, 13:1-15; Ji at 1:5-7.

As discussed in the previous section, the Ji patent discloses all of the elements of the claims of the Edery 633 patent. To the extent the examiner requires execution of the “sandboxed package” within a client sandbox and finds that the Ji patent does not disclose the “sandboxed package” claim element, claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32, and 33 would have been obvious in light of the Liu patent, because the Liu patent discloses client “sandbox” execution for Java related-technology. *See e.g.*, Liu at 2:19-41. Therefore, because the combination of the Ji and Liu patents was not considered by the examiner during prosecution, a SNQP exists and reexamination is warranted. *See In re Swanson*, 540 F.3d 1368, 1380 (Fed. Cir. 2008) (“the PTO should evaluate the context in which the reference was previously considered and the scope of the prior consideration and determine whether the reference is now being considered for a substantially different purpose”).

**E. SNQP 3 – The Ji Patent In View of the Golan Patent raises a SNQP as to Claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32, and 33 under 37 CFR 1.510(b)(1)**

The *KSR* obviousness standard demonstrates that the teachings from the Ji and Golan patents apply to the claim elements of the Edery 633 patent relating to the “sandboxed package.” One of ordinary skill in the art would be motivated to include the specific “sandbox” and Java architecture because the Ji patent, like the Golan patent, addressed network and computer security, including stopping viruses and other malicious software attacks resulting from the use of network languages like Java. *See* Ji at 1:1-39; Golan at 1:1-2:8. Importantly, the examiner set forth the bases for which the Golan patent discloses a majority of the claim elements.

---

held that “the [nonobviousness] analysis need not seek out precise teachings directed to the specific subject matter of the challenged claim, for a court can take account of the inferences and creative steps that a person of ordinary skill in the art would employ.” *Id.* at 418.

Additionally, as also outlined above in Section III.C., the Ji patent discloses the mobile protection code elements that the applicants argued were missing from the Golan patent. Therefore, the combination of the Ji patent and the Golan patent raises a SNQP, because the Ji patent discloses the precise elements the applicants argued were not present in the Golan reference. *See In re Swanson*, 540 F.3d 1368, 1380 (Fed. Cir. 2008) (“the PTO should evaluate the context in which the reference was previously considered and the scope of the prior consideration and determine whether the reference is now being considered for a substantially different purpose”).

**F. The Edery 633 Patent Is Not Entitled To A Priority Date From The Alleged Parent Continuation-In-Part Patents.**

The Edery 633 patent is a continuation of the Edery 822 patent filed on May 17, 2001. The Edery 822 patent claims the benefit of the earlier filed provisional application, no. 60/205,591 (the “591 provisional application”) filed on May 17, 2000. The Edery 822 (and hence, the Edery 633 patent) also purports to be a continuation-in-part of U.S. App. Nos. 09/551,302 and 09/539,667 (now U.S. Patent Nos. 6,480,692 and 6,804,780, respectively). However, the Edery 633 patent (and the Edery 822 patent) cannot claim priority to the 6,480,692 and 6,804,780 patents, because the Edery 633 patent describes and claims new matter not disclosed in these patents. As the MPEP explains, determination of the proper priority date is appropriate as part of a request for *ex parte* reexamination:

The statement applying the prior art may, where appropriate, point out that claims in the patent for which reexamination is requested are entitled only to the filing date of that patent and not supported by an earlier foreign or United States patent application whose filing date is claimed. For example, even where a patent is a continuing application under 35 U.S.C. 120, the effective date of some of the claims could be the filing date of the child application which resulted in the patent, because those claims were not supported in the parent application.

MPEP § 2617.

As illustrated below, although the Edery 633 patent attempts to claim priority to the 6,480,692 and/or 6,804,780 patents, the claims of the Edery 633 patent are “not supported” in those specifications. *See* U.S. Patent Nos. 6,480,692 and 6,804,780. Accordingly, the applicable priority date for purposes of the invalidity analysis cannot be earlier than May 17, 2001 (the filing date of U.S. Application No. 09/861,229 which issued as the Edery 822 patent).

The following portions of the Edery 633 patent (and the Edery 822 patent) were all “new matter” in the CIP application:

- FIG. 1a
- FIG. 1b
- FIG. 1c
- FIG. 2
- FIG. 3
- FIG. 4
- FIG. 5
- FIG. 6a
- FIG. 6b
- FIG. 7a
- FIG. 7b
- FIG. 8
- FIG. 9
- FIG. 10a
- FIG. 10b
- FIG. 11
- FIG. 12a
- FIG. 12b and
- Col. 1:55 thru Col. 24:3

Consequently, all of the descriptions relating to, among other things, mobile protection code, downloadable-information, information-destination, detection-indicators, executable code characteristic, executable file type indicators, information patterns, a sandboxed package, import address table, and a filter-driver are all new matter first introduced on May 17, 2001, with the filing of the Edery 822 patent. Similarly, all of the descriptions relating to the aforementioned, including but not limited to determining if the downloadable-information includes executable code, evaluating the significance of various executable code detection indicators, communicating mobile protection code between devices, constructing sandbox packages, constructing sandbox packages that include the downloadable file, mobile protection code and/or security policies all constitute new matter first introduced on May 17, 2001, the filing date of the Edery 822 patent.

Because the claim scope depends on the newly added material in the application, claims 1, 2, 3, 4, 5, 6, 7, 27, 28, 29, 30, 31, 32 and 33 may only receive the benefit of priority to the 2001 filing date. See, e.g., *Waldemar Link, GmbH & Co. v. Osteonics Corp.*, 32 F.3d 556, 558 (Fed. Cir. 1994). Therefore, the Ji patent, Liu patent, and Golan patent all serve as prior art over the Edery 633 patent because their disclosures predate the claimed subject matter of the Edery 633 patent.

**IV. EXPLANATION OF PERTINENCY AND MANNER OF APPLYING CITED PRIOR ART TO EVERY CLAIM FOR WHICH REEXAMINATION IS REQUESTED UNDER 37 CFR 1.510(B)(2)**

Requester submits three proposed grounds corresponding to the three SNQP discussed above as follows:

(A) Claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32 and 33 are anticipated under 35 U.S.C. § 102(e) in light of the Ji patent (SNQP 1);

(B) Claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32 and 33 are obvious under 35 U.S.C. § 103(a) in light of the Ji patent in view of Liu patent (SNQP 2); and

(C) Claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32 and 33 are obvious under 35 U.S.C. § 103(a) in light of the Ji patent in view of the Golan patent (SNQP 3).

**A. The Ji Patent**

The claim chart below details the manner of applying the Ji patent to every claim for which reexamination is requested.

As discussed above, the Ederly 633 patent is entitled to a priority date of no earlier than May 17, 2000. The Ji patent was filed on September 10, 1997. Accordingly, the Ji patent anticipates the claims under 35 U.S.C. § 102(e).

<b>Ederly 633 Patent Claim Limitations</b>	<b>The Ji Patent</b>
1. A computer processor-based method, comprising:	<p>The Ji patent discloses the preamble. Ji discloses computer implemented applications executing on a computer network. Specifically, the Abstract discloses a “network scanner for security checking of application programs [...] received over the Internet or an Intranet has both static (pre-run time) and dynamic (run time) scanning. [...] During run time at the client, the instrumented instructions are thereby monitored for security policy violations, and execution of an instruction is prevented in the event of such a violation.”</p> <p>Figure 1 discloses a server and client computer device. It is well understood that a computer includes one or more processors for executing software applications.</p>
receiving, by a computer, downloadable-information;	<p>The Ji patent discloses the step of receiving, by a computer, downloadable-information. The Ji patent discloses receiving files “(e.g. Java applets or ActiveX controls)” from the Internet at the server in Fig. 1. See also Ji patent at 3:17-23 (“Thereby in accordance with the invention a scanner (for a virus or other malicious code) provides both static and dynamic scanning for application programs, e.g. Java applets or ActiveX controls. The</p>

<b>Edery 633 Patent Claim Limitations</b>	<b>The Ji Patent</b>
<p>determining, by the computer, whether the downloadable-information includes executable code; and</p>	<p>applets or controls (hereinafter collectively referred to as applets) are conventionally received from e.g. the Internet or an Intranet at a conventional server.”).</p> <p>The Ji patent discloses the step of determining whether the downloadable-information includes executable code. The Ji patent discloses that the server includes scanning software that scans the downloaded “applet” code and instruments the applet code. Ji at 4:66-5:43. The Ji patent further discloses that non-executable code is not instrumented or scanned, specifically, “[d]ownloaded non-applets are not scanned.” Ji at 5:3-4.</p> <p>The Ji patent discloses detecting Java and ActiveX executable code (3:16-22). As the Ji patent explains (and claims in claim 1), the disclosed methods are not limited to Java and ActiveX, but apply to additional executable code types. Ji at 8:23-29.</p> <p>Additionally, the Ji patent discloses performing multiple analyses of the downloaded file to determine if the file contains executable code. In the first analysis, Ji discloses that the invention first determines what code (downloaded-information characteristic) is included in the downloaded code (downloaded-information). If the code includes Java or ActiveX code (detection indicator) then that code is of the type (executable code characteristic) identified for further processing and analysis. Ji at 4:66-5:4.</p> <p>In the second analysis, Ji discloses instrumenting the applet when “suspicious instructions” (Ji at 5:22) are identified during the scanning process, depicted in Fig. 2 and as described in 5:16-6:37. Accordingly, this second analysis identifies specific applet instructions (downloadable-information characteristics) contained in the downloaded code (downloadable-information) deemed to be “suspicious” (executable code characteristic) as determined by “a predefined set of [insecure] functions.” Ji at 5:22-23.</p> <p>Thus, the Ji patent discloses determining if the downloaded code is executable and if so, determines if it is a specific type of executable.</p>
<p>based upon the determination, transmitting from the computer mobile protection code to at least one information-destination of</p>	<p>The Ji patent discloses the step of transmitting the single JAR archive file containing the monitoring package (mobile protection code), the security (protection) policies and the applet (downloadable-information) to the intended client machine (at least one information-destination) of the applet, if</p>

<b>Edery 633 Patent Claim Limitations</b>	<b>The Ji Patent</b>
the downloadable-information, if the downloadable-information is determined to include executable code.	<p>the applet is determined to include executable code. After the applet code has been instrumented at the server, “[t]he instrumented applet is then downloaded from the server to the client (local computer), at which time the applet code is conventionally interpreted by the client Web browser and it begins to be executed. As the applet code is executed, each instrumented instruction is monitored by the Web browser using a monitor package which is part of the scanner and delivered to the client side. Upon execution, each instrumented instruction is subject to a security check. If the security policy (which has been pre-established) is violated, that particular instruction which violates the security policy is not executed, and instead a report is made and execution continues, if appropriate, with the next instruction.” Ji at 3:32-44.</p> <p>The Ji patent explains that, as depicted in Fig. 1, the security monitoring package (mobile protection code) is included with the instrumented downloadable “in a single JAR (Java archive) file format at the server <b>32</b>, and downloaded to the web browser <b>22</b> in client machine <b>14</b>.” Ji at 6:38-42.</p>
2. The method of claim 1, wherein the receiving includes monitoring received information of an information re-communicator.	<p>The Ji patent discloses monitoring received information of an information re-communicator. The Ji patent teaches that the HTTP proxy server software “identifies suspicious instructions” in received downloadables. Abstract.</p> <p>Ji at 4:66-5:3 (“Upon receipt of a particular Java applet, the HTTP proxy server 32, which is software running on server machine 20 and which has associated scanner software 26, then scans the applet and instruments it using an instrumenter 28 which is part of the scanner software 26.”).</p>
3. The method of claim 2, wherein the information re-communicator is a network server.	<p>The Ji patent discloses a method wherein the information re-communicator is a network server. It is well understood in the art that an HTTP proxy server may be installed on the network and services HTTP requests from client computers seeking access to Internet content, such as web pages.</p> <p>Ji at 4:66-5:3 (“Upon receipt of a particular Java applet, the HTTP proxy server 32, which is software running on server machine 20 and which has associated scanner software 26, then scans the applet and instruments it using an instrumenter 28 which is part of the scanner software 26.”).</p>
4. The method of claim 1, wherein the determining comprises analyzing the	The Ji patent discloses a method wherein the downloaded file (downloadable-information) is analyzed to determine if the type of file is a Java applet or ActiveX control. Ji at 3:18-20.

<b>Edery 633 Patent Claim Limitations</b>	<b>The Ji Patent</b>
downloadable-information for an included type indicator indicating an executable file type.	<p>Accordingly, where the downloaded file is an applet, the HTTP proxy scanner server receives and indicator that the file type is executable.</p> <p>Ji at 4:66-5:4 (“Upon receipt of a particular Java applet, the HTTP proxy server 32, which is software running on server machine 20 and which has associated scanner software 26, then scans the applet and instruments it using an instrumenter 28 which is part of the scanner software 26. (Downloaded non-applets are not scanned.)”).</p> <p>Additionally, as a result of the scanning process, if the result of the scan indicates the presence of at least one “suspicious instruction” the downloaded file is understood to be an executable file. Ji at 5:21-23.</p>
5. The method of claim 1, wherein the determining comprises analyzing the downloadable-information for an included type detector indicating an archive file that contains at least one executable.	<p>The Ji patent discloses analyzing the downloaded file (downloadable-information) for an included type detector indicating an archive file that contains at least one executable. The Ji patent discloses analyzing the downloaded applet for archive files:</p> <p>“An applet pre-fetcher component 38 fetches from the Internet 10 all the dependency files required by a Java class file, if they are not already packed into a JAR file. This is important because the goal is to attach the scanner monitor package to a session only once. A Java applet may contain more than one code module, or class file. Heretofore this disclosure has assumed that all the class files are packed in one JAR file and downloaded once. One monitoring package is attached to the JAR file and every instantiation of this package on the client web browser 22 marks a unique session. However, if the class files are not packed together and are downloaded on an as-needed basis during applet execution, multiple instrumentation will occur and multiple instances of the monitoring package for the same session are created on the client. This creates a problem of how to maintain information on session states. To solve this problem, the pre-fetcher 38 pre-fetches the dependency class files during the static scanning of the main applet code module. The dependency class files are (see below) instrumented once, packed together, and delivered to the client.”</p> <p>Ji at 7:8-28.</p>

<b>Edery 633 Patent Claim Limitations</b>	<b>The Ji Patent</b>
<p>6. The method of claim 1, wherein the determining comprises analyzing the downloadable-information for an included file type indicator and an information pattern corresponding to one or more information patterns that tend to be included within executable code.</p>	<p>The Ji patent discloses analyzing the applet (downloadable-information) for an included file type indicator (such as Java or ActiveX) and an information pattern corresponding to one or more information patterns that tend to be included within executable code. Ji at 3:18-20. For example, where the downloaded file is an applet, the HTTP proxy scanner server receives and indicator that the file type is executable. Ji at 4:66-5:4 (“Upon receipt of a particular Java applet, the HTTP proxy server 32, which is software running on server machine 20 and which has associated scanner software 26, then scans the applet and instruments it using an instrumenter 28 which is part of the scanner software 26. (Downloaded non-applets are not scanned.)”).</p> <p>Regarding “an information pattern corresponding to one or more information patterns that tend to be included within executable code” claim element, as a result of the scanning process, if the result of the scan indicates the presence of at least one “suspicious instruction” the downloaded file is understood to be an executable file. Ji at 5:21-23. Here, a “suspicious instruction” is understood to be an “information pattern.”</p> <p>Similarly, the use of an information pattern is disclosed by the Ji patent’s process of identifying Java class files within an archive file:</p> <p>“An applet pre-fetcher component 38 fetches from the Internet 10 all the dependency files required by a Java class file, if they are not already packed into a JAR file. This is important because the goal is to attach the scanner monitor package to a session only once.</p> <p>A Java applet may contain more than one code module, or class file. Heretofore this disclosure has assumed that all the class files are packed in one JAR file and downloaded once. One monitoring package is attached to the JAR file and every instantiation of this package on the client web browser 22 marks a unique session. However, if the class files are not packed together and are downloaded on an as-needed basis during applet execution, multiple instrumentation will occur and multiple instances of the monitoring package for the same session are created on the client. This creates a problem of how to maintain information on session states. To solve this problem, the pre-fetcher 38 pre-fetches the dependency class files</p>

Edery 633 Patent Claim Limitations	The Ji Patent
	<p>during the static scanning of the main applet code module. The dependency class files are (see below) instrumented once, packed together, and delivered to the client.”</p> <p>Ji at 7:8-28.</p>
<p>7. The method of claim 1, further comprising receiving, by the computer, one or more executable code characteristics of executable code that is capable of being executed by the information-destination, and wherein the determining is conducted in accordance with the executable code characteristics.</p>	<p>The Ji patent discloses receiving, by the HTTP proxy server (computer), one or more applets (executable code) containing potentially “suspicious” computer instructions (executable code characteristics) that are capable of being executed by the client-machine (information-destination), and wherein the determining is conducted in accordance with the computer instructions (executable code characteristics) contained in the applet and associated Java class files. Ji at 4:66-5:7 (teaching that Ji applets from non-applets and then only scans applets); <i>see also</i> 5:16-5:42 (An example of such a suspicious Java function is “Java.IO.File.list” which may list the contents of a client (local) directory 30, e.g. a directory on the client machine 14 hard disk drive. The first instruction sequence generates a call to a pre-filter function provided by the scanner 26, signaling that an insecure (suspicious) function is to be invoked. The pre-filter checks the security policy associated with the scanner 26 and decides whether this particular instruction (“call”) is allowed. The second instruction sequence generates a call to a post-filter function also provided by the scanner. It also reports the result of the call to the post-filter function. Both the pre- and post-filter functions update the session state to be used by the security policy. The static scanning and instrumentation are both performed on the HTTP proxy server 32.”).</p>
<p>28. A processor-based method, comprising:</p>	<p>The Ji patent discloses the preamble. Ji discloses computer implemented applications executing on a computer network. Specifically, the Abstract discloses a “network scanner for security checking of application programs [...] received over the Internet or an Intranet has both static (pre-run time) and dynamic (run time) scanning. [...] During run time at the client, the instrumented instructions are thereby monitored for security policy violations, and execution of an instruction is prevented in the event of such a violation.”</p> <p>Figure 1 discloses a server and client computer device. It is well understood that a computer includes one or more processors for executing software applications.</p>
<p>receiving a sandboxed package that includes mobile protection code (“MPC”) and</p>	<p>The Ji patent discloses receiving at the client computer (Downloadable destination) a single JAR archive file (sandboxed package) that includes mobile protection code and</p>

<b>Edery 633 Patent Claim Limitations</b>	<b>The Ji Patent</b>
<p>a Downloadable and one or more protection policies at a computer at a Downloadable destination;</p>	<p>one or more security policies. The Ji patent explains that, as depicted in Fig. 1, the security monitoring package (mobile protection code) is included with the instrumented downloadable “in a single JAR (Java archive) file format at the server 32, and downloaded to the web browser 22 in client machine 14.” Ji at 6:38-42.</p> <p>As the Ji patent discloses “[t]he instrumented applet is then downloaded from the server to the client (local computer), at which time the applet code is conventionally interpreted by the client Web browser and it begins to be executed. As the applet code is executed, each instrumented instruction is monitored by the Web browser using a monitor package which is part of the scanner and delivered to the client side. Upon execution, each instrumented instruction is subject to a security check. If the security policy (which has been pre-established) is violated, that particular instruction which violates the security policy is not executed, and instead a report is made and execution continues, if appropriate, with the next instruction.” Ji at 3:32-44.</p> <p>Moreover, the applet and monitor/security package are executed in a sandbox inside the web browser:</p> <p>“The monitor package contains monitoring functions that are delivered from the server 32 to the client web browser 22 with the instrumental applet and are invoked by the instrumentation code in the applet. The monitor package also creates a unique session upon instantiation. It also contains a security policy checker (supplied by security policy generator component 54) to determine whether the applet being scanned violates the security policy, given the monitoring information.</p> <p>The security policy generator component 54 generates the security checker code included in the monitor package, from a set of predefined security policies. Different clients, users, and applets may have different security policies. The security policy generator 54 may run on server machine 20 or another computer. In addition, security policies can be configured by an administrator of the system. A simple security policy is to assign different weights to monitored functions and make sure the security weight of a session does not exceed a preset threshold. A more sophisticated security policy checks the file or resource the applet is trying to access at run time and prompts the user whether to allow</p>

Edery 633 Patent Claim Limitations	The Ji Patent
	<p>the access. Hence the security policy broadly is a state machine to detect security policy violations upon attempted instruction execution.”</p> <p>Ji at 7:41-64. See also Ji at Fig. 2.</p>
<p>causing, by the MPC on the computer, one or more operations attempted by the Downloadable to be received by the MPC;</p>	<p>The Ji patent discloses the step of the monitoring package (mobile protection code) with the included security (protection) policies intercept instructions (operations) attempted by the downloadable. The Ji patent explains that “[t]he scanned (instrumented) applet, which has been digitally signed is then downloaded to the web browser 22 in the client 14. The applet is then conventionally interpreted by the web browser 22 and its instructions are executed. <b><u>The execution is monitored by the monitor package software</u></b>, also downloaded from scanner 26, in the web browser 22 in accordance with this invention for security purposes.” Ji at 5:6-13 (emphasis added); 6:43-45 (“All <b><u>the monitoring and applet code is executed in the web browser 22 in the client machine</u></b> 14.”) (emphasis added).</p> <p>The Ji patent discloses monitoring of single and multiple operations performed in series for comparison to the security policies:</p> <p>“After the code of an applet is downloaded, e.g. via the Internet to a client platform (local computer), an instance of the applet is created in the conventional Java “virtual machine” in the web browser (client) running on that local computer. Different instances of the same applet might produce different results given different inputs. A running instance of an applet is conventionally called a session; sessions are strictly run-time entities. Static scanning cannot analyze sessions because static scanning does not let the applet run. Sessions are important because an instance of an applet will often perform a series of suspicious tasks before it can be determined dangerous (i.e., in violation of the security policy). Such state information needs to be associated with the sessions. The present applet scanner thereby stops sessions instead of blocking execution of the entire applet.”</p> <p>Ji at 2:30-46.</p>
<p>receiving, by the MPC on the computer, an attempted operation of the Downloadable; and</p>	<p>The Ji patent discloses the step wherein the client machine (computer) and the monitoring package receives the instruction (operation) attempted by the Downloadable. The Ji patent explains that “[t]he scanned (instrumented) applet, which has been digitally signed is then downloaded to the web browser 22</p>

Edery 633 Patent Claim Limitations	The Ji Patent
	<p>in the client 14. <u>The applet</u> is then conventionally interpreted by the web browser 22 <u>and its instructions are executed. The execution is monitored by the monitor package software</u>, also downloaded from scanner 26, in the web browser 22 in accordance with this invention for security purposes.” Ji at 5:6-13 (emphasis added).</p>
<p>initiating, by the MPC on the computer, a protection policy corresponding to the attempted operation.</p>	<p>The Ji patent discloses the step wherein the monitoring package (mobile protection code) includes a security (protection) policies corresponding to potential actions (operations) that may be attempted by the downloadable. “A security policy defines what functions an applet needs to perform to be considered a security risk. Examples of security policies include preventing(1) applets from any file access, or (2) file access in a certain directory, or (3) creating certain Java objects. An applet scanner in accordance with the invention may allow different security policies for different clients, for different users, and for applets from different origins.” Ji at 4:47-54.</p> <p>“The monitor package contains monitoring functions that are delivered from the server 32 to the client web browser 22 with the instrumental applet and are invoked by the instrumentation code in the applet. <u>The monitor package</u> also creates a unique session upon instantiation. It also <u>contains a security policy checker</u> (supplied by security policy generator component 54) <u>to determine whether the applet being scanned violates the security policy</u>, given the monitoring information.</p> <p>The security policy generator component 54 generates the security checker code included in the monitor package, from a set of predefined security policies. Different clients, users, and applets may have different security policies.” Ji at 7:41-53 (emphasis added).</p> <p>“A simple security policy is to assign different weights to monitored functions and make sure the security weight of a session does not exceed a preset threshold. A more sophisticated security policy checks the file or resource the applet is trying to access at run time and prompts the user whether to allow the access. Hence <u>the security policy broadly is a state machine to detect security policy violations upon attempted instruction execution</u>.” Ji at 7:56-64 (emphasis added).</p>
<p>29. The method of claim 28, wherein the sandboxed</p>	<p>The Ji patent discloses the method wherein the single JAR archive file (sandboxed package) is configured such that the</p>

Edery 633 Patent Claim Limitations	The Ji Patent
<p>package is configured such that the MPC is executed first, the Downloadable is executed by the MPC and the protection policies are accessible to the MPC.</p>	<p>monitoring package (MPC) is executed first and the Downloadable is executed by the monitoring package (MPC) and the security (protection) policies are accessible to the monitoring package (MPC).</p> <p>As described in claim 28, the Ji patent discloses that “[t]he scanned (instrumented) applet, which has been digitally signed is then downloaded to the web browser 22 in the client 14. <b><u>The applet</u></b> is then conventionally interpreted by the web browser 22 <b><u>and its instructions are executed. The execution is monitored by the monitor package software</u></b>, also downloaded from scanner 26, in the web browser 22 in accordance with this invention for security purposes.” Ji at 5:6-13 (emphasis added).</p> <p>The Ji patent inherently discloses that the monitoring package (MPC) is executed prior to the downloadable because otherwise some instructions (computer operations) would not be captured thus creating a security risk.</p> <p>Additionally, “[t]he monitor package contains monitoring functions that are delivered from the server 32 to the client web browser 22 with the instrumental applet and are invoked by the instrumentation code in the applet. <b><u>The monitor package</u></b> also creates a unique session upon instantiation. It also <b><u>contains a security policy checker</u></b> (supplied by security policy generator component 54) <b><u>to determine whether the applet being scanned violates the security policy</u></b>, given the monitoring information.” Ji at 7:41-49 (emphasis added).</p>
<p>30. The method of claim 28, wherein the causing comprises modifying, by the MPC, interfaces of a corresponding downloadable to resources at the destination.</p>	<p>The Ji patent discloses the method step of using pre- and post-filter calls (modifying), by the monitoring package (MPC), interfaces of a corresponding downloadable to resources at the destination.</p> <p>“The Java instrumenter component 48 instruments the Java class files, e.g. by inserting monitoring instructions (e.g. pre and post filter calls) before and after each suspicious instruction, as described above.</p> <p>The monitor package contains monitoring functions that are delivered from the server 32 to the client web browser 22 with the instrumental applet and are invoked by the instrumentation code in the applet. The monitor package also creates a unique session upon instantiation. It also contains a security policy</p>

<b>Edery 633 Patent Claim Limitations</b>	<b>The Ji Patent</b>
	<p>checker (supplied by security policy generator component 54) to determine whether the applet being scanned violates the security policy, given the monitoring information.</p> <p>The security policy generator component 54 generates the security checker code included in the monitor package, from a set of predefined security policies. Different clients, users, and applets may have different security policies.” Ji at 7:37-53.</p> <p>“The present applet scanner thus uses applet instrumentation technology, that is, for Java applets it alters the Java applet byte code sequence during downloading of the applet to the server 32. After the Java applet byte code sequence has been downloaded, the static (pre-run time) scanning is performed on the applet by the scanner 26. If an instruction (a suspicious instruction) that calls an insecure function (as determined by a predefined set of such functions) is found during this static scanning, a first instruction sequence (pre-filter) is inserted before that instruction and a second instruction sequence (post-filter) after that instruction by the instruments.” Ji at 5:16-27.</p>
<p>31. The method of claim 30, wherein the modifying is accomplished by initiating a loading of the Downloadable, thereby causing a mobile code executor to provide and initialize the interfaces, modifying one or more interface elements to divert corresponding attempted Downloadable operations to the MPC, and initiating execution of the Downloadable.</p>	<p>The Ji patent discloses the method step wherein the pre- and post-filter instructions (modifying) is accomplished by first scanning (initiating a loading) and instrumenting (modifying one or more interface elements) the Downloadable, whereupon the web browser’s Java virtual machine (mobile code executor), after receiving the downloadable, executes the instructions, intercepting (diverting) calls to operating system resources attempted by the Downloadable.</p> <p>“The Java instrumenter component 48 instruments the Java class files, e.g. by inserting monitoring instructions (e.g. pre and post filter calls) before and after each suspicious instruction, as described above.</p> <p>The monitor package contains monitoring functions that are delivered from the server 32 to the client web browser 22 with the instrumental applet and are invoked by the instrumentation code in the applet. The monitor package also creates a unique session upon instantiation. It also contains a security policy checker (supplied by security policy generator component 54) to determine whether the applet being scanned violates the security policy, given the monitoring information.</p> <p>The security policy generator component 54 generates the</p>

Edery 633 Patent Claim Limitations	The Ji Patent
	<p>security checker code included in the monitor package, from a set of predefined security policies. Different clients, users, and applets may have different security policies.” Ji at 7:37-53.</p> <p>“The present applet scanner thus uses applet instrumentation technology, that is, for Java applets it alters the Java applet byte code sequence during downloading of the applet to the server 32. After the Java applet byte code sequence has been downloaded, the static (pre-run time) scanning is performed on the applet by the scanner 26. If an instruction (a suspicious instruction) that calls an insecure function (as determined by a predefined set of such functions) is found during this static scanning, a first instruction sequence (pre-filter) is inserted before that instruction and a second instruction sequence (post-filter) after that instruction by the instruments.” Ji at 5:16-27.</p>
32. The method of claim 30, wherein the interfaces comprise an import address table (“IAT”) of a native code executable downloadable.	The Ji patent discloses the method step wherein the interfaces comprise an import address table (“IAT”) of a native code executable downloadable. The Ji patent teaches the creation of a single JAR archive file (sandboxed package) wherein the archive file downloaded to the client computer contains all of Java classes required by the applet for execution in the Java virtual machine and allowed within the browser’s address space.
33. The method of claim 30, wherein modifying the interfaces installs a filter-driver between the downloadable and the resources.	<p>The Ji patent discloses the method step wherein the modification of the interfaces installs the monitor package (to act as a filter-driver) between the downloadable and the resources accessible to the web browser’s Java virtual machine:</p> <p>“The present applet scanner thus uses applet instrumentation technology, that is, for Java applets it alters the Java applet byte code sequence during downloading of the applet to the server 32. After the Java applet byte code sequence has been downloaded, the static (pre-run time) scanning is performed on the applet by the scanner 26. If an instruction (a suspicious instruction) that calls an insecure function (as determined by a predefined set of such functions) is found during this static scanning, a first instruction sequence (pre-filter) is inserted before that instruction and a second instruction sequence (post-filter) after that instruction by the instruments.</p> <p>An example of such a suspicious Java function is “Java.IO.File.list” which may list the contents of a client (local) directory 30, e.g. a directory on the client machine 14 hard disk drive. The first</p>

Edery 633 Patent Claim Limitations	The Ji Patent
	<p>instruction sequence generates a call to a pre-filter function provided by the scanner 26, signaling that an insecure (suspicious) function is to be invoked. The pre-filter checks the security policy associated with the scanner 26 and decides whether this particular instruction (“call”) is allowed. The second instruction sequence generates a call to a post-filter function also provided by the scanner. It also reports the result of the call to the post-filter function. Both the pre- and post-filter functions update the session state to be used by the security policy.”</p> <p>Ji at 5:16-40.</p>

### B. The Ji Patent In View Of The Liu Patent

The Ji patent expressly or inherently discloses all of the limitations in claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32 and 33 as shown above in SNQP 1.

To the extent the Examiner finds that the Ji patent does not disclose the “sandboxed package” claim element, claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32 and 33 would have been obvious in light of the Liu patent, as shown below. The Liu patent was filed on May 22, 1998. Therefore, the Liu patent is prior art over the claims of the Edery 633 patent, as discussed above.

The claim chart below addresses SNQP 2 and details the manner of applying the Ji patent in view of the Liu patent to claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32 and 33.

Edery 633 Patent Claim Limitations	The Ji Patent in View of the Liu Patent
1. A computer processor-based method, comprising:	<p>The Ji patent discloses the preamble. Ji discloses computer implemented applications executing on a computer network. Specifically, the Abstract discloses a “network scanner for security checking of application programs [...] received over the Internet or an Intranet has both static (pre-run time) and dynamic (run time) scanning. [...] During run time at the client, the instrumented instructions are thereby monitored for security policy violations, and execution of an instruction is prevented in the event of such a violation.”</p> <p>Figure 1 discloses a server and client computer device. It is well understood that a computer includes one or more processors for executing software applications.</p>
receiving, by a computer, downloadable-information;	The Ji patent discloses the step of receiving, by a computer, downloadable-information. The Ji patent discloses receiving

<b>Edery 633 Patent Claim Limitations</b>	<b>The Ji Patent in View of the Liu Patent</b>
	<p>files “(e.g. Java applets or ActiveX controls)” from the Internet at the server in Fig. 1. See also Ji patent at 3:17-23 (“Thereby in accordance with the invention a scanner (for a virus or other malicious code) provides both static and dynamic scanning for application programs, e.g. Java applets or ActiveX controls. The applets or controls (hereinafter collectively referred to as applets) are conventionally received from e.g. the Internet or an Intranet at a conventional server.”).</p>
<p>determining, by the computer, whether the downloadable-information includes executable code; and</p>	<p>The Ji patent discloses the step of determining whether the downloadable-information includes executable code. The Ji patent discloses that the server includes scanning software that scans the downloaded “applet” code and instruments the applet code. Ji at 4:66-5:43. The Ji patent further discloses that non-executable code is not instrumented or scanned, specifically, “[d]ownloaded non-applets are not scanned.” Ji at 5:3-4.</p> <p>The Ji patent discloses detecting Java and ActiveX executable code (3:16-22). As the Ji patent explains (and claims in claim 1), the disclosed methods are not limited to Java and ActiveX, but apply to additional executable code types. Ji at 8:23-29.</p> <p>Additionally, the Ji patent discloses performing multiple analyses of the downloaded file to determine if the file contains executable code. In the first analysis, Ji discloses that the invention first determines what code (downloaded-information characteristic) is included in the downloaded code (downloaded-information). If the code includes Java or ActiveX code (detection indicator) then that code is of the type (executable code characteristic) identified for further processing and analysis. Ji at 4:66-5:4.</p> <p>In the second analysis, Ji discloses instrumenting the applet when “suspicious instructions” (Ji at 5:22) are identified during the scanning process, depicted in Fig. 2 and as described in 5:16-6:37. Accordingly, this second analysis identifies specific applet instructions (downloadable-information characteristics) contained in the downloaded code (downloadable-information) deemed to be “suspicious” (executable code characteristic) as determined by “a predefined set of [insecure] functions.” Ji at 5:22-23.</p> <p>Thus, the Ji patent discloses determining if the downloaded code is executable and if so, determines if it is a specific type of executable.</p>

<b>Edery 633 Patent Claim Limitations</b>	<b>The Ji Patent in View of the Liu Patent</b>
<p>based upon the determination, transmitting from the computer mobile protection code to at least one information-destination of the downloadable-information, if the downloadable-information is determined to include executable code.</p>	<p>The Ji patent discloses the step of transmitting the single JAR archive file containing the monitoring package (mobile protection code), the security (protection) policies and the applet (downloadable-information) to the intended client machine (at least one information-destination) of the applet, if the applet is determined to include executable code. After the applet code has been instrumented at the server, “[t]he instrumented applet is then downloaded from the server to the client (local computer), at which time the applet code is conventionally interpreted by the client Web browser and it begins to be executed. As the applet code is executed, each instrumented instruction is monitored by the Web browser using a monitor package which is part of the scanner and delivered to the client side. Upon execution, each instrumented instruction is subject to a security check. If the security policy (which has been pre-established) is violated, that particular instruction which violates the security policy is not executed, and instead a report is made and execution continues, if appropriate, with the next instruction.” Ji at 3:32-44.</p> <p>The Ji patent explains that, as depicted in Fig. 1, the security monitoring package (mobile protection code) is included with the instrumented downloadable “in a single JAR (Java archive) file format at the server <b>32</b>, and downloaded to the web browser <b>22</b> in client machine <b>14</b>.” Ji at 6:38-42.</p>
<p>2. The method of claim 1, wherein the receiving includes monitoring received information of an information re-communicator.</p>	<p>The Ji patent discloses monitoring received information of an information re-communicator. The Ji patent teaches that the HTTP proxy server software “identifies suspicious instructions” in received downloadables. Abstract.</p> <p>Ji at 4:66-5:3 (“Upon receipt of a particular Java applet, the HTTP proxy server 32, which is software running on server machine 20 and which has associated scanner software 26, then scans the applet and instruments it using an instrumenter 28 which is part of the scanner software 26.”).</p>
<p>3. The method of claim 2, wherein the information re-communicator is a network server.</p>	<p>The Ji patent discloses a method wherein the information re-communicator is a network server. It is well understood in the art that an HTTP proxy server may be installed on the network and services HTTP requests from client computers seeking access to Internet content, such as web pages.</p> <p>Ji at 4:66-5:3 (“Upon receipt of a particular Java applet, the HTTP proxy server 32, which is software running on server machine 20 and which has associated scanner software 26, then</p>

Edery 633 Patent Claim Limitations	The Ji Patent in View of the Liu Patent
	scans the applet and instruments it using an instrumenter 28 which is part of the scanner software 26.”).
4. The method of claim 1, wherein the determining comprises analyzing the downloadable-information for an included type indicator indicating an executable file type.	<p>The Ji patent discloses a method wherein the downloaded file (downloadable-information) is analyzed to determine if the type of file is a Java applet or ActiveX control. Ji at 3:18-20. Accordingly, where the downloaded file is an applet, the HTTP proxy scanner server receives and indicator that the file type is executable.</p> <p>Ji at 4:66-5:4 (“Upon receipt of a particular Java applet, the HTTP proxy server 32, which is software running on server machine 20 and which has associated scanner software 26, then scans the applet and instruments it using an instrumenter 28 which is part of the scanner software 26. (Downloaded non-applets are not scanned.)”).</p> <p>Additionally, as a result of the scanning process, if the result of the scan indicates the presence of at least one “suspicious instruction” the downloaded file is understood to be an executable file. Ji at 5:21-23.</p>
5. The method of claim 1, wherein the determining comprises analyzing the downloadable-information for an included type detector indicating an archive file that contains at least one executable.	<p>The Ji patent discloses analyzing the downloaded file (downloadable-information) for an included type detector indicating an archive file that contains at least one executable. The Ji patent discloses analyzing the downloaded applet for archive files:</p> <p>“An applet pre-fetcher component 38 fetches from the Internet 10 all the dependency files required by a Java class file, if they are not already packed into a JAR file. This is important because the goal is to attach the scanner monitor package to a session only once.</p> <p>A Java applet may contain more than one code module, or class file. Heretofore this disclosure has assumed that all the class files are packed in one JAR file and downloaded once. One monitoring package is attached to the JAR file and every instantiation of this package on the client web browser 22 marks a unique session. However, if the class files are not packed together and are downloaded on an as-needed basis during applet execution, multiple instrumentation will occur and multiple instances of the monitoring package for the same session are created on the client. This creates a problem of how to maintain information on session states. To solve this problem, the pre-fetcher 38 pre-fetches the dependency class files</p>

<b>Edery 633 Patent Claim Limitations</b>	<b>The Ji Patent in View of the Liu Patent</b>
	<p>during the static scanning of the main applet code module. The dependency class files are (see below) instrumented once, packed together, and delivered to the client.”</p> <p>Ji at 7:8-28.</p>
<p>6. The method of claim 1, wherein the determining comprises analyzing the downloadable-information for an included file type indicator and an information pattern corresponding to one or more information patterns that tend to be included within executable code.</p>	<p>The Ji patent discloses analyzing the applet (downloadable-information) for an included file type indicator (such as Java or ActiveX) and an information pattern corresponding to one or more information patterns that tend to be included within executable code. Ji at 3:18-20. For example, where the downloaded file is an applet, the HTTP proxy scanner server receives and indicator that the file type is executable. Ji at 4:66-5:4 (“Upon receipt of a particular Java applet, the HTTP proxy server 32, which is software running on server machine 20 and which has associated scanner software 26, then scans the applet and instruments it using an instrumenter 28 which is part of the scanner software 26. (Downloaded non-applets are not scanned.)”).</p> <p>Regarding “an information pattern corresponding to one or more information patterns that tend to be included within executable code” claim element, as a result of the scanning process, if the result of the scan indicates the presence of at least one “suspicious instruction” the downloaded file is understood to be an executable file. Ji at 5:21-23. Here, a “suspicious instruction” is understood to be an “information pattern.”</p> <p>Similarly, the use of an information pattern is disclosed by the Ji patent’s process of identifying Java class files within an archive file:</p> <p>“An applet pre-fetcher component 38 fetches from the Internet 10 all the dependency files required by a Java class file, if they are not already packed into a JAR file. This is important because the goal is to attach the scanner monitor package to a session only once.</p> <p>A Java applet may contain more than one code module, or class file. Heretofore this disclosure has assumed that all the class files are packed in one JAR file and downloaded once. One monitoring package is attached to the JAR file and every instantiation of this package on the client web browser 22 marks a unique session. However, if the class files are not packed together and are downloaded on an as-needed basis during applet execution, multiple instrumentation will</p>

Edery 633 Patent Claim Limitations	The Ji Patent in View of the Liu Patent
	<p>occur and multiple instances of the monitoring package for the same session are created on the client. This creates a problem of how to maintain information on session states. To solve this problem, the pre-fetcher 38 pre-fetches the dependency class files during the static scanning of the main applet code module. The dependency class files are (see below) instrumented once, packed together, and delivered to the client.”</p> <p>Ji at 7:8-28.</p>
<p>7. The method of claim 1, further comprising receiving, by the computer, one or more executable code characteristics of executable code that is capable of being executed by the information-destination, and wherein the determining is conducted in accordance with the executable code characteristics.</p>	<p>The Ji patent discloses receiving, by the HTTP proxy server (computer), one or more applets (executable code) containing potentially “suspicious” computer instructions (executable code characteristics) that are capable of being executed by the client-machine (information-destination), and wherein the determining is conducted in accordance with the computer instructions (executable code characteristics) contained in the applet and associated Java class files. Ji at 4:66-5:7 (teaching that Ji applets from non-applets and then only scans applets); <i>see also</i> 5:16-5:42 (An example of such a suspicious Java function is “Java.IO.File.list” which may list the contents of a client (local) directory 30, e.g. a directory on the client machine 14 hard disk drive. The first instruction sequence generates a call to a pre-filter function provided by the scanner 26, signaling that an insecure (suspicious) function is to be invoked. The pre-filter checks the security policy associated with the scanner 26 and decides whether this particular instruction (“call”) is allowed. The second instruction sequence generates a call to a post-filter function also provided by the scanner. It also reports the result of the call to the post-filter function. Both the pre- and post-filter functions update the session state to be used by the security policy. The static scanning and instrumentation are both performed on the HTTP proxy server 32.”).</p>
<p>28. A processor-based method, comprising:</p>	<p>The Ji patent discloses the preamble. Ji discloses computer implemented applications executing on a computer network. Specifically, the Abstract discloses a “network scanner for security checking of application programs [...] received over the Internet or an Intranet has both static (pre-run time) and dynamic (run time) scanning. [...] During run time at the client, the instrumented instructions are thereby monitored for security policy violations, and execution of an instruction is prevented in the event of such a violation.”</p> <p>Figure 1 discloses a server and client computer device. It is</p>

<b>Edery 633 Patent Claim Limitations</b>	<b>The Ji Patent in View of the Liu Patent</b>
	well understood that a computer includes one or more processors for executing software applications.
receiving a sandboxed package that includes mobile protection code (“MPC”) and a Downloadable and one or more protection policies at a computer at a Downloadable destination;	<p>The Ji patent discloses receiving at the client computer (Downloadable destination) a single JAR archive file (sandboxed package) that includes mobile protection code and one or more security policies. The Ji patent explains that, as depicted in Fig. 1, the security monitoring package (mobile protection code) is included with the instrumented downloadable “in a single JAR (Java archive) file format at the server 32, and downloaded to the web browser 22 in client machine 14.” Ji at 6:38-42.</p> <p>As the Ji patent discloses “[t]he instrumented applet is then downloaded from the server to the client (local computer), at which time the applet code is conventionally interpreted by the client Web browser and it begins to be executed. As the applet code is executed, each instrumented instruction is monitored by the Web browser using a monitor package which is part of the scanner and delivered to the client side. Upon execution, each instrumented instruction is subject to a security check. If the security policy (which has been pre-established) is violated, that particular instruction which violates the security policy is not executed, and instead a report is made and execution continues, if appropriate, with the next instruction.” Ji at 3:32-44.</p> <p>Moreover, the applet and monitor/security package are executed in a sandbox inside the web browser:</p> <p>“The monitor package contains monitoring functions that are delivered from the server 32 to the client web browser 22 with the instrumental applet and are invoked by the instrumentation code in the applet. The monitor package also creates a unique session upon instantiation. It also contains a security policy checker (supplied by security policy generator component 54) to determine whether the applet being scanned violates the security policy, given the monitoring information.</p> <p>The security policy generator component 54 generates the security checker code included in the monitor package, from a set of predefined security policies. Different clients, users, and applets may have different security policies. The security policy generator 54 may run on server machine 20 or another computer. In addition, security policies can be configured by an administrator of the system. A simple security policy is</p>

Edery 633 Patent Claim Limitations	The Ji Patent in View of the Liu Patent
	<p>to assign different weights to monitored functions and make sure the security weight of a session does not exceed a preset threshold. A more sophisticated security policy checks the file or resource the applet is trying to access at run time and prompts the user whether to allow the access. Hence the security policy broadly is a state machine to detect security policy violations upon attempted instruction execution.”</p> <p>Ji at 7:41-64. See also Ji at Fig. 2.</p> <p><b><u>OBVIOUSNESS</u></b></p> <p>Regarding the “sandboxed package” limitation, the claims of the Edery 633 patent do not require the execution of the “sandboxed package” in a “sandbox” on the client computer (“client-side ‘sandbox’ execution”). However, if the Examiner determines (or patentee later argues) that the claims do require execution of the “sandboxed package” within a client “sandbox” and the Examiner decides that the Ji patent does not sufficiently disclose client “sandbox” execution; the Liu patent discloses client “sandbox” execution for Java related-technology. Specifically, the Liu patent discloses the “Java sandbox” architecture:</p> <p>“Given that such ultimately foreign program code is downloaded for local execution, there are inherent security issues that could arise and, therefore, which have been addressed and circumvented in advance within the Java architecture. The Java architecture includes security features that prevent such downloaded programs from interfering with the user’s private or non-network resources. Referred to as the ‘Java sandbox’, the Java architecture prevents an untrusted or potentially malicious applet (downloaded to the local end system from a remote web server) from reading, writing, or executing private resources, such as the local hard drive. Among other security features, the Java language is a typesafe language, which does not allow pointers to read or write to arbitrary memory locations. In addition, prior to execution of an incoming applet, the applet is run through a Java bytecode verifier, which examines the bytecode for potentially illegal commands, such that only legal applets get executed by the JVM at the local end system. See, e.g., Java Security Whitepaper, available at the Sun web sites <a href="http://java.sun.com">java.sun.com</a> and <a href="http://javasoft.com">javasoft.com</a>; A. Tanenbaum, Computer Networks (Prentice-Hall, 3d ed. 1996), at 718-20; D. 40 Flanagan,</p>

Edery 633 Patent Claim Limitations	The Ji Patent in View of the Liu Patent
	<p>Java in a Nutshell, (O'Reilly, 2d ed. 1997), at 7, 139-43.” Liu at 2:19-41.</p> <p>As such, to the extent the Ji patent’s disclosure regarding execution within the client’s web browser of the single JAR archive file, communicated from the Ji patent’s server 20 to client machine 14 for execution within the web browser’s Java virtual machine (described above) departs from the “sandbox” and Java architecture disclosed in the Liu patent, it would have been obvious to combine client-side “sandbox” execution in the Liu patent with the specific Java architecture disclosed in the Ji patent. One of ordinary skill in the art would be motivated to include the specific “sandbox” and Java architecture because the Liu patent, like the Ji patent, addressed network and computer security, including stopping virus and other malicious software attacks resulting from the use of network languages like Java. Liu at 1:40-2:58, 4:28-50, 13:1-15; Ji at 1:5-7.</p> <p>Thus, the Ji patent in view of the Liu patent discloses the client-side “sandbox” execution.</p>
<p>causing, by the MPC on the computer, one or more operations attempted by the Downloadable to be received by the MPC;</p>	<p>The Ji patent discloses the step of the monitoring package (mobile protection code) with the included security (protection) policies intercept instructions (operations) attempted by the downloadable. The Ji patent explains that “[t]he scanned (instrumented) applet, which has been digitally signed is then downloaded to the web browser 22 in the client 14. The applet is then conventionally interpreted by the web browser 22 and its instructions are executed. <b><u>The execution is monitored by the monitor package software</u></b>, also downloaded from scanner 26, in the web browser 22 in accordance with this invention for security purposes.” Ji at 5:6-13 (emphasis added); 6:43-45 (“All <b><u>the monitoring and applet code is executed in the web browser 22 in the client machine</u></b> 14.”) (emphasis added).</p> <p>The Ji patent discloses monitoring of single and multiple operations performed in series for comparison to the security policies:</p> <p>“After the code of an applet is downloaded, e.g. via the Internet to a client platform (local computer), an instance of the applet is created in the conventional Java “virtual machine” in the web browser (client) running on that local computer. Different instances of the same applet might produce different results given different inputs. A running instance of an applet is</p>

Edery 633 Patent Claim Limitations	The Ji Patent in View of the Liu Patent
	<p>conventionally called a session; sessions are strictly run-time entities. Static scanning cannot analyze sessions because static scanning does not let the applet run. Sessions are important because an instance of an applet will often perform a series of suspicious tasks before it can be determined dangerous (i.e., in violation of the security policy). Such state information needs to be associated with the sessions. The present applet scanner thereby stops sessions instead of blocking execution of the entire applet.”</p> <p>Ji at 2:30-46.</p>
<p>receiving, by the MPC on the computer, an attempted operation of the Downloadable; and</p>	<p>The Ji patent discloses the step wherein the client machine (computer) and the monitoring package receives the instruction (operation) attempted by the Downloadable. The Ji patent explains that “[t]he scanned (instrumented) applet, which has been digitally signed is then downloaded to the web browser 22 in the client 14. <b>The applet</b> is then conventionally interpreted by the web browser 22 <b>and its instructions are executed. The execution is monitored by the monitor package software</b>, also downloaded from scanner 26, in the web browser 22 in accordance with this invention for security purposes.” Ji at 5:6-13 (emphasis added).</p>
<p>initiating, by the MPC on the computer, a protection policy corresponding to the attempted operation.</p>	<p>The Ji patent discloses the step wherein the monitoring package (mobile protection code) includes a security (protection) policies corresponding to potential actions (operations) that may be attempted by the downloadable. “A security policy defines what functions an applet needs to perform to be considered a security risk. Examples of security policies include preventing(1) applets from any file access, or (2) file access in a certain directory, or (3) creating certain Java objects. An applet scanner in accordance with the invention may allow different security policies for different clients, for different users, and for applets from different origins.” Ji at 4:47-54.</p> <p>“The monitor package contains monitoring functions that are delivered from the server 32 to the client web browser 22 with the instrumental applet and are invoked by the instrumentation code in the applet. <b>The monitor package</b> also creates a unique session upon instantiation. It also <b>contains a security policy checker</b> (supplied by security policy generator component 54) <b>to determine whether the applet being scanned violates the security policy</b>, given the monitoring information.</p> <p>The security policy generator component 54 generates the security checker code included in the monitor package, from a</p>

Edery 633 Patent Claim Limitations	The Ji Patent in View of the Liu Patent
	<p>set of predefined security policies. Different clients, users, and applets may have different security policies.” Ji at 7:41-53 (emphasis added).</p> <p>“A simple security policy is to assign different weights to monitored functions and make sure the security weight of a session does not exceed a preset threshold. A more sophisticated security policy checks the file or resource the applet is trying to access at run time and prompts the user whether to allow the access. Hence <b><u>the security policy broadly is a state machine to detect security policy violations upon attempted instruction execution.</u></b>” Ji at 7:56-64 (emphasis added).</p>

### C. The Ji Patent In View Of The Golan Patent

The Ji patent expressly or inherently discloses all of the limitations in claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32 and 33 as shown above in SNQP 1.

To the extent the Examiner finds that the Ji patent does not disclose the “sandboxed package” claim element, claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32 and 33 would have been obvious in light of the Golan patent as shown below. The Golan patent was filed on March 27, 1997. Therefore, the Golan patent is prior art over the claims of the Edery 633 patent, as discussed above.

The claim chart below addresses SNQP 3 and details the manner of applying the Ji patent in view of the Golan patent to claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32 and 33.

Edery 633 Patent Claim Limitations	The Ji Patent in View of the Golan Patent
1. A computer processor-based method, comprising:	<p>The Ji patent discloses the preamble. Ji discloses computer implemented applications executing on a computer network. Specifically, the Abstract discloses a “network scanner for security checking of application programs [...] received over the Internet or an Intranet has both static (pre-run time) and dynamic (run time) scanning. [...] During run time at the client, the instrumented instructions are thereby monitored for security policy violations, and execution of an instruction is prevented in the event of such a violation.”</p> <p>Figure 1 discloses a server and client computer device. It is well understood that a computer includes one or more processors for executing software applications.</p>

<b>Edery 633 Patent Claim Limitations</b>	<b>The Ji Patent in View of the Golan Patent</b>
receiving, by a computer, downloadable-information;	<p>The Ji patent discloses the step of receiving, by a computer, downloadable-information. The Ji patent discloses receiving files “(e.g. Java applets or ActiveX controls)” from the Internet at the server in Fig. 1. See also Ji patent at 3:17-23 (“Thereby in accordance with the invention a scanner (for a virus or other malicious code) provides both static and dynamic scanning for application programs, e.g. Java applets or ActiveX controls. The applets or controls (hereinafter collectively referred to as applets) are conventionally received from e.g. the Internet or an Intranet at a conventional server.”).</p>
determining, by the computer, whether the downloadable-information includes executable code; and	<p>The Ji patent discloses the step of determining whether the downloadable-information includes executable code. The Ji patent discloses that the server includes scanning software that scans the downloaded “applet” code and instruments the applet code. Ji at 4:66-5:43. The Ji patent further discloses that non-executable code is not instrumented or scanned, specifically, “[d]ownloaded non-applets are not scanned.” Ji at 5:3-4.</p> <p>The Ji patent discloses detecting Java and ActiveX executable code (3:16-22). As the Ji patent explains (and claims in claim 1), the disclosed methods are not limited to Java and ActiveX, but apply to additional executable code types. Ji at 8:23-29.</p> <p>Additionally, the Ji patent discloses performing multiple analyses of the downloaded file to determine if the file contains executable code. In the first analysis, Ji discloses that the invention first determines what code (downloaded-information characteristic) is included in the downloaded code (downloaded-information). If the code includes Java or ActiveX code (detection indicator) then that code is of the type (executable code characteristic) identified for further processing and analysis. Ji at 4:66-5:4.</p> <p>In the second analysis, Ji discloses instrumenting the applet when “suspicious instructions” (Ji at 5:22) are identified during the scanning process, depicted in Fig. 2 and as described in 5:16-6:37. Accordingly, this second analysis identifies specific applet instructions (downloadable-information characteristics) contained in the downloaded code (downloadable-information) deemed to be “suspicious” (executable code characteristic) as determined by “a predefined set of [insecure] functions.” Ji at 5:22-23.</p> <p>Thus, the Ji patent discloses determining if the downloaded</p>

Edery 633 Patent Claim Limitations	The Ji Patent in View of the Golan Patent
	code is executable and if so, determines if it is a specific type of executable.
based upon the determination, transmitting from the computer mobile protection code to at least one information-destination of the downloadable-information, if the downloadable-information is determined to include executable code.	<p>The Ji patent discloses the step of transmitting the single JAR archive file containing the monitoring package (mobile protection code), the security (protection) policies and the applet (downloadable-information) to the intended client machine (at least one information-destination) of the applet, if the applet is determined to include executable code. After the applet code has been instrumented at the server, “[t]he instrumented applet is then downloaded from the server to the client (local computer), at which time the applet code is conventionally interpreted by the client Web browser and it begins to be executed. As the applet code is executed, each instrumented instruction is monitored by the Web browser using a monitor package which is part of the scanner and delivered to the client side. Upon execution, each instrumented instruction is subject to a security check. If the security policy (which has been pre-established) is violated, that particular instruction which violates the security policy is not executed, and instead a report is made and execution continues, if appropriate, with the next instruction.” Ji at 3:32-44.</p> <p>The Ji patent explains that, as depicted in Fig. 1, the security monitoring package (mobile protection code) is included with the instrumented downloadable “in a single JAR (Java archive) file format at the server <b>32</b>, and downloaded to the web browser <b>22</b> in client machine <b>14</b>.” Ji at 6:38-42.</p>
2. The method of claim 1, wherein the receiving includes monitoring received information of an information re-communicator.	<p>The Ji patent discloses monitoring received information of an information re-communicator. The Ji patent teaches that the HTTP proxy server software “identifies suspicious instructions” in received downloadables. Abstract.</p> <p>Ji at 4:66-5:3 (“Upon receipt of a particular Java applet, the HTTP proxy server 32, which is software running on server machine 20 and which has associated scanner software 26, then scans the applet and instruments it using an instrumenter 28 which is part of the scanner software 26.”).</p>
3. The method of claim 2, wherein the information re-communicator is a network server.	<p>The Ji patent discloses a method wherein the information re-communicator is a network server. It is well understood in the art that an HTTP proxy server may be installed on the network and services HTTP requests from client computers seeking access to Internet content, such as web pages.</p> <p>Ji at 4:66-5:3 (“Upon receipt of a particular Java applet, the</p>

Edery 633 Patent Claim Limitations	The Ji Patent in View of the Golan Patent
	<p>HTTP proxy server 32, which is software running on server machine 20 and which has associated scanner software 26, then scans the applet and instruments it using an instrumenter 28 which is part of the scanner software 26.”).</p>
<p>4. The method of claim 1, wherein the determining comprises analyzing the downloadable-information for an included type indicator indicating an executable file type.</p>	<p>The Ji patent discloses a method wherein the downloaded file (downloadable-information) is analyzed to determine if the type of file is a Java applet or ActiveX control. Ji at 3:18-20. Accordingly, where the downloaded file is an applet, the HTTP proxy scanner server receives and indicator that the file type is executable.</p> <p>Ji at 4:66-5:4 (“Upon receipt of a particular Java applet, the HTTP proxy server 32, which is software running on server machine 20 and which has associated scanner software 26, then scans the applet and instruments it using an instrumenter 28 which is part of the scanner software 26. (Downloaded non-applets are not scanned.)”).</p> <p>Additionally, as a result of the scanning process, if the result of the scan indicates the presence of at least one “suspicious instruction” the downloaded file is understood to be an executable file. Ji at 5:21-23.</p>
<p>5. The method of claim 1, wherein the determining comprises analyzing the downloadable-information for an included type detector indicating an archive file that contains at least one executable.</p>	<p>The Ji patent discloses analyzing the downloaded file (downloadable-information) for an included type detector indicating an archive file that contains at least one executable. The Ji patent discloses analyzing the downloaded applet for archive files:</p> <p>“An applet pre-fetcher component 38 fetches from the Internet 10 all the dependency files required by a Java class file, if they are not already packed into a JAR file. This is important because the goal is to attach the scanner monitor package to a session only once.</p> <p>A Java applet may contain more than one code module, or class file. Heretofore this disclosure has assumed that all the class files are packed in one JAR file and downloaded once. One monitoring package is attached to the JAR file and every instantiation of this package on the client web browser 22 marks a unique session. However, if the class files are not packed together and are downloaded on an as-needed basis during applet execution, multiple instrumentation will occur and multiple instances of the monitoring package for the same session are created on the client. This creates a problem of how to maintain information</p>

<b>Edery 633 Patent Claim Limitations</b>	<b>The Ji Patent in View of the Golan Patent</b>
	<p>on session states. To solve this problem, the pre-fetcher 38 pre-fetches the dependency class files during the static scanning of the main applet code module. The dependency class files are (see below) instrumented once, packed together, and delivered to the client.”</p> <p>Ji at 7:8-28.</p>
<p>6. The method of claim 1, wherein the determining comprises analyzing the downloadable-information for an included file type indicator and an information pattern corresponding to one or more information patterns that tend to be included within executable code.</p>	<p>The Ji patent discloses analyzing the applet (downloadable-information) for an included file type indicator (such as Java or ActiveX) and an information pattern corresponding to one or more information patterns that tend to be included within executable code. Ji at 3:18-20. For example, where the downloaded file is an applet, the HTTP proxy scanner server receives and indicator that the file type is executable. Ji at 4:66-5:4 (“Upon receipt of a particular Java applet, the HTTP proxy server 32, which is software running on server machine 20 and which has associated scanner software 26, then scans the applet and instruments it using an instrumenter 28 which is part of the scanner software 26. (Downloaded non-applets are not scanned.)”).</p> <p>Regarding “an information pattern corresponding to one or more information patterns that tend to be included within executable code” claim element, as a result of the scanning process, if the result of the scan indicates the presence of at least one “suspicious instruction” the downloaded file is understood to be an executable file. Ji at 5:21-23. Here, a “suspicious instruction” is understood to be an “information pattern.”</p> <p>Similarly, the use of an information pattern is disclosed by the Ji patent’s process of identifying Java class files within an archive file:</p> <p>“An applet pre-fetcher component 38 fetches from the Internet 10 all the dependency files required by a Java class file, if they are not already packed into a JAR file. This is important because the goal is to attach the scanner monitor package to a session only once.</p> <p>A Java applet may contain more than one code module, or class file. Heretofore this disclosure has assumed that all the class files are packed in one JAR file and downloaded once. One monitoring package is attached to the JAR file and every instantiation of this package on the client web browser 22 marks a unique session. However, if the class files are not packed</p>

Edery 633 Patent Claim Limitations	The Ji Patent in View of the Golan Patent
	<p>together and are downloaded on an as-needed basis during applet execution, multiple instrumentation will occur and multiple instances of the monitoring package for the same session are created on the client. This creates a problem of how to maintain information on session states. To solve this problem, the pre-fetcher 38 pre-fetches the dependency class files during the static scanning of the main applet code module. The dependency class files are (see below) instrumented once, packed together, and delivered to the client.”</p> <p>Ji at 7:8-28.</p>
<p>7. The method of claim 1, further comprising receiving, by the computer, one or more executable code characteristics of executable code that is capable of being executed by the information-destination, and wherein the determining is conducted in accordance with the executable code characteristics.</p>	<p>The Ji patent discloses receiving, by the HTTP proxy server (computer), one or more applets (executable code) containing potentially “suspicious” computer instructions (executable code characteristics) that are capable of being executed by the client-machine (information-destination), and wherein the determining is conducted in accordance with the computer instructions (executable code characteristics) contained in the applet and associated Java class files. Ji at 4:66-5:7 (teaching that Ji applets from non-applets and then only scans applets); <i>see also</i> 5:16-5:42 (An example of such a suspicious Java function is “Java.IO.File.list” which may list the contents of a client (local) directory 30, e.g. a directory on the client machine 14 hard disk drive. The first instruction sequence generates a call to a pre-filter function provided by the scanner 26, signaling that an insecure (suspicious) function is to be invoked. The pre-filter checks the security policy associated with the scanner 26 and decides whether this particular instruction (“call”) is allowed. The second instruction sequence generates a call to a post-filter function also provided by the scanner. It also reports the result of the call to the post-filter function. Both the pre- and post-filter functions update the session state to be used by the security policy. The static scanning and instrumentation are both performed on the HTTP proxy server 32.”).</p>
<p>28. A processor-based method, comprising:</p>	<p>The Ji patent discloses the preamble. Ji discloses computer implemented applications executing on a computer network. Specifically, the Abstract discloses a “network scanner for security checking of application programs [...] received over the Internet or an Intranet has both static (pre-run time) and dynamic (run time) scanning. [...] During run time at the client, the instrumented instructions are thereby monitored for security policy violations, and execution of an instruction is prevented in the event of such a violation.”</p>

<b>Edery 633 Patent Claim Limitations</b>	<b>The Ji Patent in View of the Golan Patent</b>
	<p>Figure 1 discloses a server and client computer device. It is well understood that a computer includes one or more processors for executing software applications.</p>
<p>receiving a sandboxed package that includes mobile protection code (“MPC”) and a Downloadable and one or more protection policies at a computer at a Downloadable destination;</p>	<p>The Ji patent discloses receiving at the client computer (Downloadable destination) a single JAR archive file (sandboxed package) that includes mobile protection code and one or more security policies. The Ji patent explains that, as depicted in Fig. 1, the security monitoring package (mobile protection code) is included with the instrumented downloadable “in a single JAR (Java archive) file format at the server 32, and downloaded to the web browser 22 in client machine 14.” Ji at 6:38-42.</p> <p>As the Ji patent discloses “[t]he instrumented applet is then downloaded from the server to the client (local computer), at which time the applet code is conventionally interpreted by the client Web browser and it begins to be executed. As the applet code is executed, each instrumented instruction is monitored by the Web browser using a monitor package which is part of the scanner and delivered to the client side. Upon execution, each instrumented instruction is subject to a security check. If the security policy (which has been pre-established) is violated, that particular instruction which violates the security policy is not executed, and instead a report is made and execution continues, if appropriate, with the next instruction.” Ji at 3:32-44.</p> <p>Moreover, the applet and monitor/security package are executed in a sandbox inside the web browser:</p> <p>“The monitor package contains monitoring functions that are delivered from the server 32 to the client web browser 22 with the instrumental applet and are invoked by the instrumentation code in the applet. The monitor package also creates a unique session upon instantiation. It also contains a security policy checker (supplied by security policy generator component 54) to determine whether the applet being scanned violates the security policy, given the monitoring information.</p> <p>The security policy generator component 54 generates the security checker code included in the monitor package, from a set of predefined security policies. Different clients, users, and applets may have different security policies. The security policy generator 54 may run on server machine 20 or another computer. In</p>

Edery 633 Patent Claim Limitations	The Ji Patent in View of the Golan Patent
	<p>addition, security policies can be configured by an administrator of the system. A simple security policy is to assign different weights to monitored functions and make sure the security weight of a session does not exceed a preset threshold. A more sophisticated security policy checks the file or resource the applet is trying to access at run time and prompts the user whether to allow the access. Hence the security policy broadly is a state machine to detect security policy violations upon attempted instruction execution.”</p> <p>Ji at 7:41-64. See also Ji at Fig. 2.</p> <p><b><u>OBVIOUSNESS</u></b></p> <p>Regarding the “sandboxed package” limitation, the claims of the Edery 633 patent do not require the execution of the “sandboxed package” in a “sandbox” on the client computer (“client-side ‘sandbox’ execution”). However, if the Examiner determines (or patentee later argues) that the claims do require execution of the “sandboxed package” within a client “sandbox” and the Examiner decides that the Ji patent does not sufficiently disclose client “sandbox” execution; the Golan patent discloses client “sandbox” execution for Java related-technology. Specifically, the Golan patent discloses the “Java sandbox” architecture:</p> <p>“The present invention is a method of creating a secure sandbox within which a plurality of downloaded software components can execute in a secure manner. The software components can be of any type, e.g., Java, ActiveX, Netscape plugin, etc. The invention implements a security monitor that is injected to the address space of an arbitrary monitored application such as a Web browser, e.g., Internet Explorer, Netscape Navigator, etc. The monitored application then executes in a secure mode in which every software component downloaded executes in a secure sandbox. The security monitor detects when such a download of a software component occurs and is operative to create the sandbox around it before it is permitted to execute. If the software component attempts to commit an action that breaches security, it halts the software component’s execution and issues a warning to the user.” Golan at 2:12-27.</p> <p>Additionally, during prosecution of Edery 633 patent, the patentee argued that Golan disclosed a sandbox execution within a web browser. June 22, 2005 Resp. to OA at 1 (“Golan</p>

Edery 633 Patent Claim Limitations	The Ji Patent in View of the Golan Patent
	<p>describes a security monitor that creates a sandbox around an application, such as a web browser, and controls the behavior of software components, such as ActiveX controls, that are operative in conjunction with the application.”).</p> <p>As such, to the extent the Ji patent’s disclosure regarding execution within the client’s web browser of the single JAR archive file, communicated from the Ji patent’s server 20 to client machine 14 for execution within the web browser’s Java virtual machine (described above) departs from the “sandbox” architecture disclosed in the Golan patent, it would have been obvious to combine client-side “sandbox” execution in the Golan patent with the specific Java architecture disclosed in the Ji patent. One of ordinary skill in the art would be motivated to include the specific “sandbox” and Java architecture because the Golan patent, like the Ji patent, addressed network and computer security, including stopping virus and other malicious software attacks resulting from the use of network languages like Java and ActiveX. Golan at 1:10-2:9; Ji at 1:5-7.</p> <p>Thus, the Ji patent in view of the Golan patent discloses the client-side “sandbox” execution.</p>
<p>causing, by the MPC on the computer, one or more operations attempted by the Downloadable to be received by the MPC;</p>	<p>The Ji patent discloses the step of the monitoring package (mobile protection code) with the included security (protection) policies intercept instructions (operations) attempted by the downloadable. The Ji patent explains that “[t]he scanned (instrumented) applet, which has been digitally signed is then downloaded to the web browser 22 in the client 14. The applet is then conventionally interpreted by the web browser 22 and its instructions are executed. <b><u>The execution is monitored by the monitor package software</u></b>, also downloaded from scanner 26, in the web browser 22 in accordance with this invention for security purposes.” Ji at 5:6-13 (emphasis added); 6:43-45 (“All <b><u>the monitoring and applet code is executed in the web browser 22 in the client machine 14.</u></b>”) (emphasis added).</p> <p>The Ji patent discloses monitoring of single and multiple operations performed in series for comparison to the security policies:</p> <p>“After the code of an applet is downloaded, e.g. via the Internet to a client platform (local computer), an instance of the applet is created in the conventional Java “virtual machine” in the web browser (client) running on that local computer. Different instances of</p>

Edery 633 Patent Claim Limitations	The Ji Patent in View of the Golan Patent
	<p>the same applet might produce different results given different inputs. A running instance of an applet is conventionally called a session; sessions are strictly run-time entities. Static scanning cannot analyze sessions because static scanning does not let the applet run. Sessions are important because an instance of an applet will often perform a series of suspicious tasks before it can be determined dangerous (i.e., in violation of the security policy). Such state information needs to be associated with the sessions. The present applet scanner thereby stops sessions instead of blocking execution of the entire applet.”</p> <p>Ji at 2:30-46.</p>
<p>receiving, by the MPC on the computer, an attempted operation of the Downloadable; and</p>	<p>The Ji patent discloses the step wherein the client machine (computer) and the monitoring package receives the instruction (operation) attempted by the Downloadable. The Ji patent explains that “[t]he scanned (instrumented) applet, which has been digitally signed is then downloaded to the web browser 22 in the client 14. <b><u>The applet</u></b> is then conventionally interpreted by the web browser 22 <b><u>and its instructions are executed. The execution is monitored by the monitor package software,</u></b> also downloaded from scanner 26, in the web browser 22 in accordance with this invention for security purposes.” Ji at 5:6-13 (emphasis added).</p>
<p>initiating, by the MPC on the computer, a protection policy corresponding to the attempted operation.</p>	<p>The Ji patent discloses the step wherein the monitoring package (mobile protection code) includes a security (protection) policies corresponding to potential actions (operations) that may be attempted by the downloadable. “A security policy defines what functions an applet needs to perform to be considered a security risk. Examples of security policies include preventing(1) applets from any file access, or (2) file access in a certain directory, or (3) creating certain Java objects. An applet scanner in accordance with the invention may allow different security policies for different clients, for different users, and for applets from different origins.” Ji at 4:47-54.</p> <p>“The monitor package contains monitoring functions that are delivered from the server 32 to the client web browser 22 with the instrumental applet and are invoked by the instrumentation code in the applet. <b><u>The monitor package</u></b> also creates a unique session upon instantiation. It also <b><u>contains a security policy checker</u></b> (supplied by security policy generator component 54) <b><u>to determine whether the applet being scanned violates the security policy,</u></b> given the monitoring information. The security</p>

Edery 633 Patent Claim Limitations	The Ji Patent in View of the Golan Patent
	<p>policy generator component 54 generates the security checker code included in the monitor package, from a set of predefined security policies. Different clients, users, and applets may have different security policies.” Ji at 7:41-53 (emphasis added).</p> <p>“A simple security policy is to assign different weights to monitored functions and make sure the security weight of a session does not exceed a preset threshold. A more sophisticated security policy checks the file or resource the applet is trying to access at run time and prompts the user whether to allow the access. Hence <b><u>the security policy broadly is a state machine to detect security policy violations upon attempted instruction execution.</u></b>” Ji at 7:56-64 (emphasis added).</p>

## V. CONCLUSION

Based on the above remarks, it is respectfully submitted that substantial new questions of patentability have been raised with respect to claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32 and 33 of the Edery 633 patent. Therefore, reexamination of the claims 1, 2, 3, 4, 5, 6, 7, 28, 29, 30, 31, 32 and 33 is respectfully requested.

Respectfully submitted,

Dated: October 7, 2013

/Ryan W. Cobb/  
 Ryan W. Cobb  
 Reg. No. 64,598  
 Attorney for Requestor

DLA PIPER LLP (US)  
 401 B Street, Suite 1700  
 San Diego, CA 92101  
 ryan.cobb@dlapiper.com  
 (619) 699-2700  
 (619) 699-2701 (fax)

---

**Acknowledgement Receipt**

---

The USPTO has received your submission at **14:16:05** Eastern Time on **07-OCT-2013** by Deposit Account: **[REDACTED]**.

\$ **12000** fee paid by e-Filer via *RAM* with Confirmation Number: 448.

You have also pre-authorized the following payments from your USPTO Deposit Account:

Charge any Additional Fees required under 37 C.F.R. Section 1.17 (Patent application and reexamination processing fees)

---

**eFiled Application Information**





---

EFS ID	17055002
Application Number	90013016
Confirmation Number	9521
Title	MALICIOUS MOBILE CODE RUNTIME MONITORING SYSTEM AND METHODS
First Named Inventor	Yigal Mordechai Edery
Customer Number or Correspondence Address	Ryan W. Cobb DLA Piper LLP (US) 401 B Street Suite 1700 San Diego CA 92101 US 619-699-2635 ryan.cobb@dlapiper.com
Filed By	Ryan Cobb
Attorney Docket Number	382984-000006
Filing Date	
Receipt Date	07-OCT-2013
Application Type	Reexam (Third Party)

---

**Application Details**

---

Submitted Files	Page Count	Document Description	File Size	Warnings
01PTOTransmittal633.PDF	2	Transmittal of New Application	52428 bytes	 PASS
02USP7647633.pdf	28	Copy of patent for which reexamination is requested	1900820 bytes	 PASS
03RequestForReexam633.pdf	46	Receipt of Orig. Ex Parte Request by Third Party	360310 bytes	 PASS
04IDS633.PDF	2	Reexam - Info Disclosure Statement Filed by 3rd Party	63340 bytes	 PASS
		Reexam Miscellaneous	652403	

05USP5983348.pdf	9	Incoming Letter	bytes	◆ PASS
06USP6058482.pdf	14	Reexam Miscellaneous Incoming Letter	1040854 bytes	◆ PASS
07USP5974549.pdf	26	Reexam Miscellaneous Incoming Letter	1447474 bytes	◆ PASS
08CertificateofService.pdf	1	Reexam Miscellaneous Incoming Letter	129979 bytes	◆ PASS
fee-info.pdf	2	Fee Worksheet (SB06)	29718 bytes	◆ PASS

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

#### New Applications Under 35 U.S.C. 111

If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.



#### National Stage of an International Application under 35 U.S.C. 371

If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

#### New International Application Filed with the USPTO as a Receiving Office

If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

#### *If you need help:*

- To ask questions about Patent e-Filing, or to suggest improvements to the online system, or report technical problems, please call the Patent Electronic Business Center at (866) 217-9197  (toll free) or send email to [EBC@uspto.gov](mailto:EBC@uspto.gov).
- Send general questions about USPTO programs to the [USPTO Contact Center \(UCC\)](#).
- For general questions regarding a petition, or requirements for filing a petition, contact the Office of Petitions Help Desk at 1 800-786-9199 .